

Lecture 12

Brad McNeney

2017-04-05

Load packages

```
library(ggplot2)  
library(dplyr)
```

```
##  
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':  
##  
##      filter, lag
```

```
## The following objects are masked from 'package:base':  
##  
##      intersect, setdiff, setequal, union
```

```
library(broom)  
Auto <- read.csv("Auto.csv", stringsAsFactors = FALSE)  
Auto <- na.omit(Auto)
```

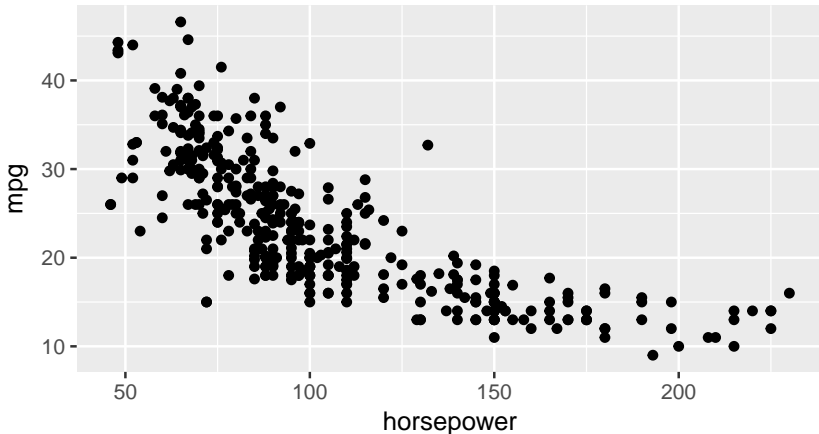
Cross-validation

- ▶ Reference: Chapter 5 of An Introduction to Statistical Learning with Applications in R Gareth James, Daniela Witten, Trevor Hastie and Robert Tibshirani

Example data

- Data on car mileage, engine size, car weight, for 392 cars made between 1970 and 1982.

```
ggplot(Auto, aes(x=horsepower, y=mpg)) + geom_point()
```



Predict mpg with polynomials in horsepower

```
afit <- lm(mpg ~ horsepower + I(horsepower^2) + I(horsepower^3),  
           data=Auto)  
tidy(afit)
```

##	term	estimate	std.error	statistic	p.value
## 1	(Intercept)	6.068478e+01	4.563446e+00	13.2980167	1.649066e-33
## 2	horsepower	-5.688501e-01	1.179222e-01	-4.8239460	2.025851e-06
## 3	I(horsepower^2)	2.079011e-03	9.479300e-04	2.1932117	2.888499e-02
## 4	I(horsepower^3)	-2.146626e-06	2.378265e-06	-0.9026016	3.672973e-01

- Suggests we need only the linear and quadratic terms.

The validation method

- ▶ Randomly split the data into a “training” set and “test” set.
- ▶ Fit the model of each polynomial degree to the training set and use it to predict observations in the test set.
- ▶ Judge the fit of each model by the mean squared error in the test set, defined as $MSE = \frac{1}{n_{test}} \sum_i (y_i - \hat{y}_i)^2$, where n_{test} is the size of the test set.
- ▶ Choose the degree with the lowest MSE.

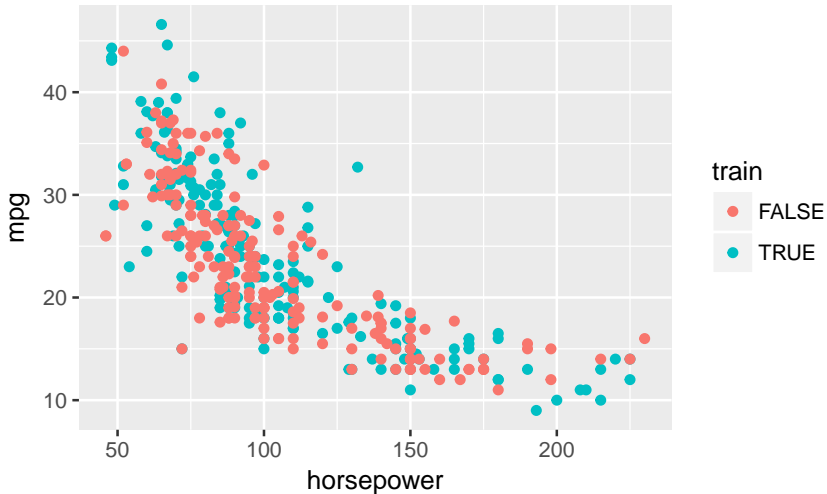
Split the data

- ▶ Several ways to do this.
 - ▶ E.G., permute the n rows and take the first $n/2$ to be the training set and the remaining $n/2$ to be the test set.

```
n <- nrow(Auto)
inds <- 1:n
set.seed(42)
pAuto <- Auto[sample(inds),]
traininds <- 1:(n/2)
trainset <- pAuto[traininds,]
testset <- pAuto[-traininds,]
```

Split the data

```
pAuto <- mutate(pAuto, train = (inds %in% traininds))  
ggplot(pAuto, aes(x=horsepower, y=mpg, color=train)) + geom_point()
```



Validation method on the Auto data

```
validate <- function(dat, ndegrees) {  
  n <- nrow(dat)  
  pdat <- dat[sample(1:n),]  
  traininds <- 1:(n/2)  
  trainset <- pdat[traininds,]  
  testset <- pdat[-traininds,]  
  MSE <- vector(length=ndegrees)  
  for(degree in 1:ndegrees) {  
    fit <- lm(mpg ~ poly(horsepower, degree), data=trainset)  
    ptest <- predict(fit, newdata=testset)  
    MSE[degree] <- mean((testset$mpg - ptest)^2)  
  }  
  data.frame(degree = 1:ndegrees, MSE = MSE)  
}
```

Validation method on Auto

```
validate(Auto,10)
```

##	degree	MSE
## 1	1	23.55361
## 2	2	17.16342
## 3	3	17.11455
## 4	4	17.48105
## 5	5	17.27775
## 6	6	18.69245
## 7	7	18.63700
## 8	8	18.98228
## 9	9	18.77249
## 10	10	19.08101

Problems with the validation method

- ▶ Splitting reduces the size of the sample used to fit the model.
- ▶ The tuning parameter that gives the best MSE can vary by split.

```
validate(Auto,10)
```

##	degree	MSE
## 1	1	26.61893
## 2	2	21.15002
## 3	3	21.11454
## 4	4	21.11325
## 5	5	20.45691
## 6	6	20.24181
## 7	7	20.03533
## 8	8	20.06858
## 9	9	20.46171
## 10	10	20.73248

Cross validation

- ▶ Split the data into k “folds”
 - ▶ Typical values of k are 5 and 10
- ▶ Leave out the first fold as a test set and train on the remaining folds.
- ▶ Repeat, leaving out each fold in turn.
- ▶ Report the average MSE over the folds.

Cross validation on the Auto data

```
createFolds <- function(n,k) { cut(1:n,breaks=k,labels=FALSE) }
crossValidate <- function(dat,ndegrees,k=10) {
  n <- nrow(dat)
  pdat <- dat[sample(1:n),]
  folds <- createFolds(n,k)
  MSE <- matrix(NA,nrow=ndegrees,ncol=k)
  for(degree in 1:ndegrees) {
    for(fold in 1:k) {
      trainset <- pdat[folds != fold,]
      testset <- pdat[folds == fold,]
      fit <- lm(mpg ~ poly(horsepower,degree),data=trainset)
      ptest <- predict(fit,newdata=testset)
      MSE[degree,fold] <- mean((testset$mpg - ptest)^2)
    }
  }
  data.frame(degree = 1:ndegrees, MSE = rowMeans(MSE))
}
```

Cross validation on Auto

```
crossValidate(Auto, ndegrees=10)
```

##	degree	MSE
## 1	1	24.32184
## 2	2	19.19603
## 3	3	19.25960
## 4	4	19.28926
## 5	5	18.87781
## 6	6	18.79886
## 7	7	18.60278
## 8	8	18.81882
## 9	9	18.96773
## 10	10	19.14339

The caret package

- ▶ The caret package is supposed to contain many functions to help with training and fitting models.
 - ▶ Have never used it myself.